



Early Identification of Implicit Requirements with the COTIR Approach using Common Sense, Ontology and Text Mining

Onyeka Emebo and Aparna S. Varde

MAY 2016

TECHNICAL REPORT ON FULLBRIGHT SCHOLARSHIP VISIT

Visiting Ph.D. Student: Mr. Onyeka Emebo

Advisor: Dr. Aparna Varde

DEPARTMENT OF COMPUTER SCIENCE

MONTCLAIR STATE UNIVERSITY

MONTCLAIR, NJ, USA

Abstract

The ability of a system to meet its requirements is a strong determinant of success. Thus effective Software Requirements Specification (SRS) is crucial. Explicit Requirements are well-defined needs for a system to execute. IMplicit Requirements (IMRs) are assumed needs that a system is expected to fulfill though not elicited during requirements gathering. Studies have shown that a major factor in the failure of software systems is the presence of unhandled IMRs. Since relevance of IMRs is important for efficient system functionality, there are methods developed to aid the identification and management of IMRs. In this research, we emphasize that commonsense knowledge, in the field of Knowledge Representation in AI, would be useful to automatically identify and manage IMRs. This research is aimed at identifying the sources of IMRs and also proposing an automated support tool for managing IMRs within an organizational context. Since this is found to be a present gap in practice, our work makes a contribution here. We propose a novel approach called COTIR (Commonsense, Ontology and Text mining for Implicit Requirements) to identify and manage IMRs. As the name implies, COTIR is based on an integrated framework of three core technologies: commonsense knowledge (CSK), text mining and ontology. We claim that discovery and handling of unknown and non-elicited requirements would reduce risks and costs in software development.

Keywords- Commonsense Knowledge, Implicit Requirements, Ontology, Requirements Engineering, Text Mining

1. Introduction

The challenge of identifying and managing implicit requirements has developed to be a crucial subject in the field of Requirements Engineering. In [7] it was stated that “When critical knowledge, goals, expectations or assumptions of key stakeholders remain hidden or unshared then poor requirements and poor systems are a likely, and costly, consequence.” With the relevance of implicit requirements (IMRs) being identified and related to the efficient functionality of any developed system, there have been different proposals as well as practical methodologies developed to aid the identification and management of IMRs. Commonsense Knowledge (CSK) is an area that involves making a computer or another machine understand basic facts intuitively as a human would. It is an area in the realm of Knowledge Representation (KR) which involves paradigms for adequate depiction of knowledge in Artificial Intelligence (AI). The area of CSK is being researched for its use in identification and capturing of implicit requirements.

Since AI is aimed at enabling machines solve problems like humans, there is a need for commonsense knowledge in AI systems to enhance problem-solving. This not only involves storing what most humans know but also the application of that knowledge [8]. In software engineering, the development of systems must meet the needs of the intended user. However the very fact that CSK is common, not all knowledge and requirements that entail common sense, will be captured or expressed by the expected user. As Polanyi describes “We know more than we can tell”. It is therefore the responsibility of the software developer to capture as well as manage the unexpressed requirements in the development of a suitable and satisfactory system. The application of commonsense knowledge can thus improve the identification as well as management of IMRs. Commonsense knowledge (CSK) is defined in [3] as a collection of simple facts about people and everyday life, such as "Things fall down, not up", and "People eat breakfast in the morning". In [7], the authors describe CSK as a tremendous amount and variety of knowledge of default

assumptions about the world, which is shared by (possibly a group of) people and seems so fundamental and obvious that it usually does not explicitly appear in people's communications. Hence, CSK is mainly characterized by its implicitness.

From the literature, it is observed that a number of reasons have caused the emergence of implicit requirements some of which include: i) a software organization develops a product in a new domain and ii) there is a knowledge gap between developers and stakeholders due to the existence of implicit knowledge.

Given this background, we claim that CSK will aid in the identification of IMRs useful for domain-specific knowledge bases. This will be useful for storing domain concepts, relations and instances for onward use in domain related processing, knowledge reuse and discovery. Thus we build an automated IMR support tool based on our proposed framework for managing IMRs using common sense knowledge, ontology and text mining.

2. Background of Relevant Technologies

In this section, an overview of the core technologies and concepts relevant to CSK, ontology and text mining / natural language processing is presented. This is useful in order to understand our proposed framework later.

2.1 Commonsense Knowledge

Commonsense Knowledge (CSK) according to [17] is a tremendous amount and variety of knowledge of default assumptions about the world, which is shared by people and seems so obvious that it usually does

not explicitly appear in communications. Some characteristics of CSK as identified in the literature are as follows:

- *Share*: A group of people possess and share CSK.
- *Fundamentality*: People have a good understanding of CSK that they tend to take CSK for granted.
- *Implicitness*: People more often don't mention or document CSK explicitly since others also know it.
- *Large-Scale*: CSK is highly diversified and in large quantity.
- *Open-Domain*: CSK is broad in nature covering all aspects of our daily life rather than a specific domain.
- *Default*: CSK are default assumptions about typical cases in everyday life, so most of them might not always be correct.

Previous work on common sense knowledge includes the seminal projects Cyc [9] and WordNet [5], ConceptNet [20], Webchild [31] and the work by [14] and [24]. Cyc has compiled complex assertions such as every human has exactly one father and exactly one mother. WordNet has manually organized nouns and adjectives into lexical classes, with careful distinction between words and word senses. ConceptNet is probably the largest repository of common sense assertions about the world, covering relations such as hasProperty, usedFor, madeOf, motivatedByGoal, etc. Tandon et al. [14] automatically compiled millions of triples of the form <noun relation adjective> by mining n-gram corpora. Lebani & Pianta [24] proposed encoding additional lexical relations for commonsense knowledge into WordNet.

WebChild contains triples that connect nouns with adjectives via fine-grained relations like hasShape, hasTaste, evokesEmotion, etc. This can be further used to extract common sense concepts for building domain specific knowledge bases [37]. Such knowledge is useful to capture subtle human reasoning.

2.2 Ontology

According to [12], ontology is a “formal explicit specification of shared conceptualization” and involves “classifications of the existing concepts”. Ontologies provide a formal representation of knowledge and the relationships between concepts of a domain [18]. They are used in Requirements Specification to guide unambiguous and formal specification of requirements, particularly in expressing concepts, relations and business rules of domain models with varying degrees of formalization and precision [26]. Formally an Ontology structure O can be defined as:

$$O = \{C, R, A^o\}$$

Where:

1. C is a set whose elements are called *concepts*.
2. $R \subseteq C \times C$ is a set whose elements are called *relations*. For $r = (c_1, c_2) \in R$, it is written $r(c_1) = c_2$.
3. A^o is a set of axioms on O .

In Requirements Engineering (RE), ontology can be used for: 1) describing requirements specification documents and 2) to formally represent requirements knowledge [10]. Moreover, ontologies can be used to support requirements management and traceability, which help software engineers understand the relations and dependencies among various software artifacts. Ontology can also be used as resources of domain knowledge, especially in a specific application domain. By using such ontology, several kinds of semantic processing can be achieved in requirements analysis [31]. In this work, ontology is considered a valid solution approach, because it has the potential to facilitate formalized semantic description of relevant domain knowledge for identification and management of IMRs.

2.3 Text Mining and Natural Language Processing

Text mining is the process of analyzing text to extract information that is useful for particular purposes [32]. Natural Language Processing (NLP) generally refers to a range of theoretically motivated computational techniques for analyzing and representing naturally occurring texts [7]. The core purpose of NLP techniques is to realize human-like language processing for a range of tasks or applications [8]. The core NLP models used in this research are part-of-speech (POS) tagging and sentence parsers [7]. POS tagging involves marking up the words in a text as corresponding to a particular part of speech, based on both its definition, as well as its context. In addition, sentence parsers transform text into a data structure (called a parse tree), which provides insight into the grammatical structure and implied hierarchy of the input text [7].

NLP is used for our purpose in analysis of requirements statements to gain an understanding of similarities that exist between requirements and/or identify a potential basis for analogy. NLP in combination with ontology enables the extraction of useful knowledge from natural language requirements documents for the early identification and management of potential IMRs.

3. Literature Survey with Related Work

Different researchers have proposed various ways for identification of IMRs. In [23], the authors present a methodology which builds on modularizing variable feature requirements with aspects, using explicit join relationships for their integration semantics, modeling the commonality and the variability of the product line in a single aspectual model, describing details of the variability including variability constraints

in tabular form, and visualizing variability constraints graphically. This methodology also shows that previously product line requirement knowledge was identified and handled as implicit by relying on the expertise of involved stakeholders. However this model only emphasizes on the documentation of implicit knowledge and not its extraction or management. In [22], the researchers present a descriptive view to provide the technical analyst with the relevance of implicit knowledge. This also aids in improving quality of requirements and their management. This study highlights evolving tools and techniques to improve the management of requirements information through automatic trace recovery. It also discovers the presence of tacit knowledge from tracking of presuppositions and non-provenance requirements. Further, it detects nocuous ambiguity in requirements documents that imply the potential for misinterpretation. This study however, lays primary emphasis on implicit knowledge and does not highlight other sources of implicitness such as ambiguity.

Using requirements reuse for discovery and management of IMRs has been covered by a few studies. A study that draws on an analogy-making approach in managing IMRs is presented in [21]. This study proposes a system that uses semantic case-based reasoning for managing IMR. The model of a tool that facilitates the management of IMRs through analogy-based requirements reuse of previously known IMRs is presented. The system comprises two major components: semantic matching for requirements similarity and analogy-based reasoning for fine-grained cross domain reuse. This approach ensures the discovery, structured documentation, proper prioritization, and evolution of IMR, which is expected to improve the overall success of software development processes. However, as of now, this has not been adopted in a practical form. The work in [25] presents a model for computing similarities between requirements specifications to promote their analogical reuse. Hence, requirement reuse is based on the detection on analogies in specifications. This model is based on the assumption of semantic modeling abstractions

including classification generalization and attribution. The semantics of these abstractions enable the employment of general criteria for detecting analogies between specifications without relying on other special knowledge. Different specification models are supported simultaneously. The similarity model which is relatively tolerant to incompleteness of specifications improves as the semantic content is enriched and copes well with large scale problems. Although the identification of analogies in requirements is essential, this study does not discuss the subject for the management of requirements. The research in [15] presents the application of rules derived for the elicitation of implicitly expressed requirements in IT ecosystems. By introducing rules into the infrastructure of the ecosystem which is being observed for adherence by agents interacting in the system, deviations from these rules can be harnessed for finding potential candidates for new or changed requirements. These deviations are then processed using techniques like data mining and pattern recognition and then forwarded to requirements engineers for review. These implicitly expressed requirements are then leveraged to identify actual changes in the needs of the users of the systems, thereby enabling further advancements.

A method to highlight requirements potentially based on implicit or implicit-like knowledge is proposed in [2]. The identification is made possible by examining the origin of each requirement, effectively showing the source material that contributes to it. It is demonstrated that a semantic-level comparison enabling technique is appropriate for this purpose. The work helps to identify the source of explicit requirements based on tacit-like knowledge but it does not specifically categorize tacit requirement and its management. Also, in MaTREx [22], a brief review and interpretation of the literature on implicit knowledge useful for requirement engineering is presented. The authors describe a number of techniques that offer analysts the means to reason the effect of implicit knowledge and improve quality of requirements and their management. The focus of their work is on evolving tools and techniques to improve the management of

requirements information through automatic trace recovery, discovering presence of tacit knowledge from tracking of presuppositions, non-provenance requirements etc. However, MaTREx still deals more with handling implicit knowledge.

Previous work on common sense knowledge includes the Cyc project [9] with a goal to codify millions of pieces of common sense knowledge in machine readable form that enable a machine to perform human-like reasoning on such knowledge. Another source is WordNet [5] in which nouns and adjectives are manually organized into lexical classes, furthermore a distinction is made between words and word senses; yet its limitation is that there are no semantic relationships between the nouns and adjectives with the exception of extremely sparse attribute relations. Another prominent collection of commonsense is ConceptNet [20], which consists mainly of crowd sourced information. ConceptNet is the outcome of Open Mind Common Sense (OMCS) [6]. OMCS has distributed this CSK gathering task across general public on the Web. Through the OMCS website, volunteer contributors can enter CSK in natural language or even evaluate CSK entered by others. Common sense knowledge is useful in harnessing human judgment and subtle aspects of reasoning that are very intuitive and is therefore helpful in related research such as [38] where CSK has been used to simulate human reasoning in mining social media data in the context of an application in urban planning.

4. The Proposed COTIR Approach

We propose an approach called COTIR: Commonsense, Ontology and Text Mining for Implicit Requirements. As the name implies, this integrates three core technologies: text mining, commonsense and ontology. The framework of the COTIR approach is presented in Figure 1. The core system functionalities

are depicted as rectangular boxes, while the logic, data and knowledge artifacts that enable core system functionalities are represented using oval boxes. A detailed description of this framework is given below.

4.1 Data Preprocessing

A preprocessed requirements document is the input to the framework. Preprocessing is a manual procedure that ensures that the requirements document is in the required format acceptable for use in the system. This entails extraction of boundary sentences from the requirements document and further representing images, figures, tables etc. in its equivalent textual format.

4.2 NL Processor

The NL processor component facilitates the processing of natural language requirements for the process that enables feature extractor. The core natural language processing operations implemented in the architecture are as follows:

- *Sentence selection:* This helps in splitting the requirements statements into sentences for onward processing.
- *Tokenization:* This further splits the requirements sentences into tokens. These tokens are usually words, punctuation, numbers, etc.
- *Part-of-speech (POS) tagging:* This classifies the tokens (words) into parts of speech such as noun, verb, adjective and pronoun.
- *Entity detection:* The process of dividing a text in syntactically correlated parts of words, like noun groups, verb groups, but does not specify their internal structure, nor their role in the main sentence.

- *Parsing*: This creates the syntax tree which represents the grammatical structure of requirements statements, in order to determine phrases, subjects, objects and predicates.

The Apache OpenNLP library [27] for natural language processing has been used to implement all NLP operations in the COTIR approach.

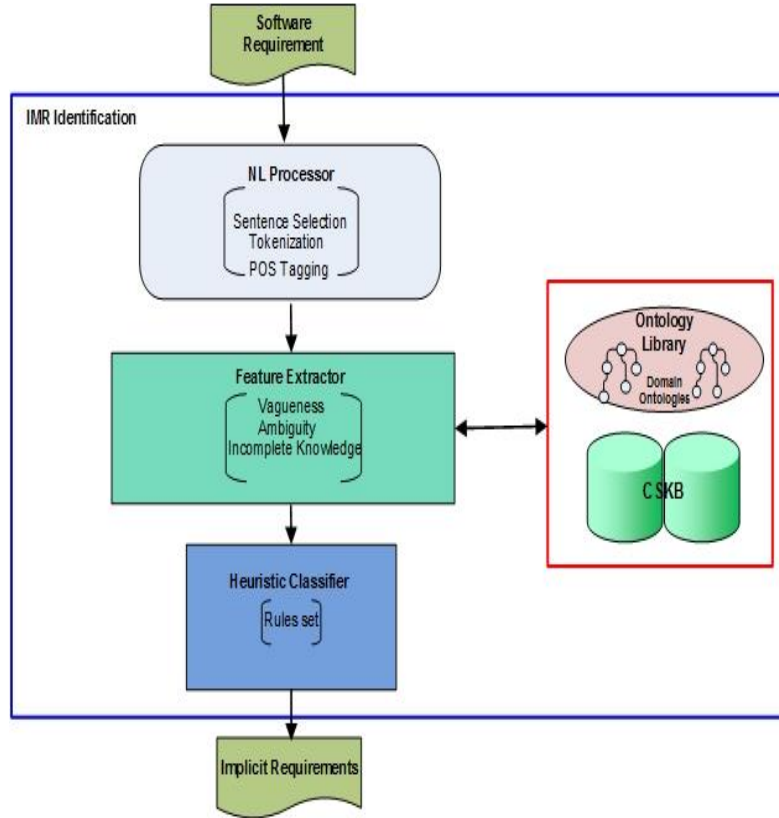


Figure 1. The Proposed COTIR Approach

4.3 Ontology Library

The ontology library and CSKB both make up the knowledge model of our framework. The ontology library serves as a storehouse for the various domain ontologies (.owl/.rdf). The domain ontologies are those that have been developed for specific purpose or those of business rules. The ontology library is implemented using Java Protégé 4.1 ontology API.

4.4 Common Sense Knowledge Base (CSKB)

The common sense knowledge bases of WebChild and domain-specific KBs are used for enhanced identification of IMR for specific domain.

4.5 Feature Extractor

The feature extractor heuristic gives the underlying assumptions for identifying potential sources of IMR in a requirement document. Due to semantic features on which natural language text exist and by taking into account previous work done [11, 13, 16, 19, 28], the following characteristic features underline the significant aspects in a piece of text in terms of surface understanding that could possibly make a requirement implicit:

- Ambiguity such as structural and lexical ambiguity.
- Presence of vague words and phrases such as “to a great extent”.
- Vague imprecise verbs such as “supported”, “handled”, “processed”, or “rejected”
- Presence of weak phrases such as “normally”, “generally”.
- Incomplete knowledge.

5. COTIR Use and Evaluation

The framework of our proposed COTIR approach illustrated in Figure 1 is used to develop a tool by the same name for the management of implicit requirements based on text mining, ontology and common sense knowledge. We describe the use of this COTIR tool for managing IMRs, followed by a demo snapshot of the tool and its evaluation.

5.1 Use of the COTIR Tool

The process of using the IMR tool developed in this work is as follows.

- *Step 1:* Preprocess the source documents to get the requirements into text file format devoid of graphics, images and tables.
- *Step 2:* Select the existing CSKB to be used for the identification of IMRs.
- *Step 3:* Import the requirement documents and domain ontology into the tool environment.
- *Step 4:* Click on the “analyze” function of the tool to allow the feature extractor identify potential sources of IMRs in the requirement document.
- *Step 5:* See the potential IMRs that are identified as well as their recommended possible explicit requirements.
- *Step 6:* Seek the expert opinion on the IMRs; the experts could approve or disapprove the recommendations by adding / removing the recommendations through the tool.

A demo snapshot of this IMR Support tool is shown in Figure 2. This is self-explanatory and can be used with the steps listed above. For evaluation of this IMR tool developed, we conduct an assessment of its performance using requirements specification. The objectives of the evaluation are as follows: (1) to assess the performance of the tool by human experts, (2) to determine its usefulness as a support tool for implicit requirements management within an organization, and (3) to identify areas of possible improvement in the tool.

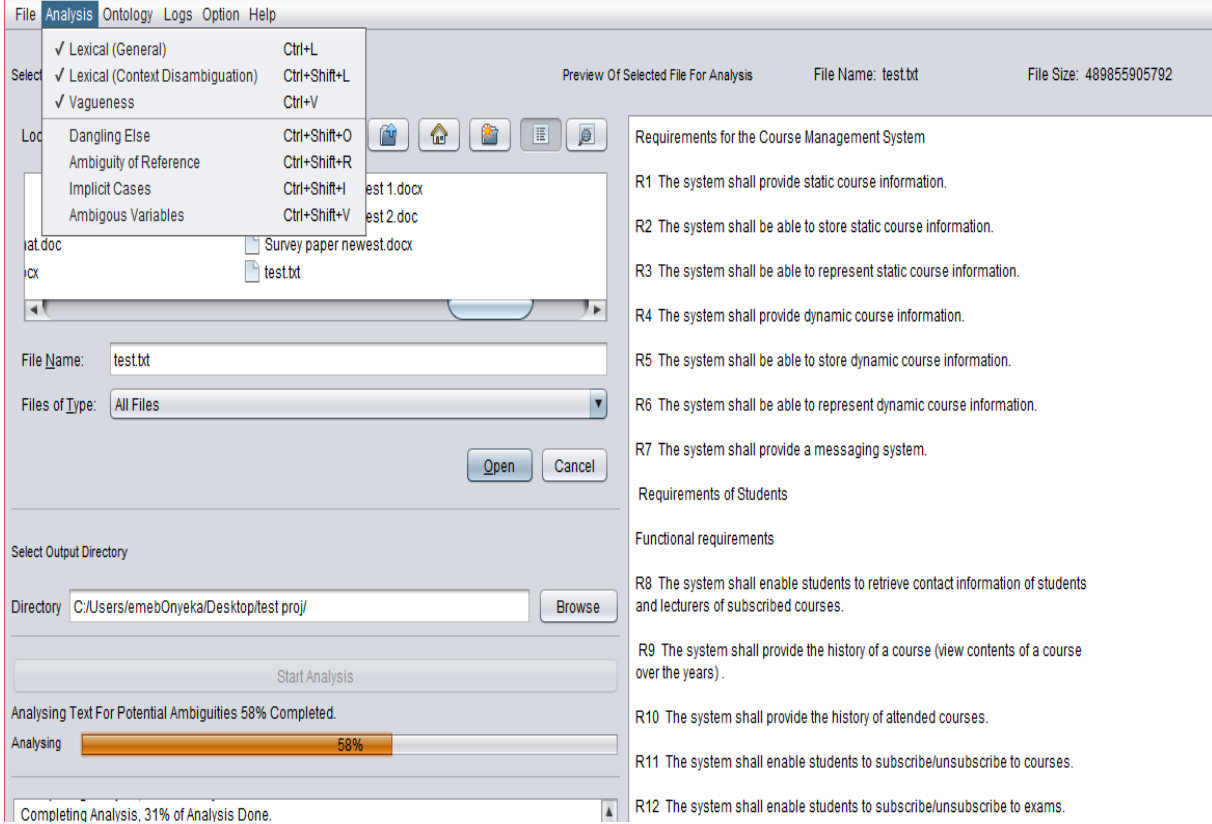


Figure 2. Demo Snapshot of COTIR Tool

5.2 Performance Evaluation Procedure

The evaluation makes use of the following sets of requirements specification from targeted applications:

i) Course Management System [33], ii) Embedded Monitoring Project [34], and iii) Tactical Control System (TCS) requirements [35]. These three requirements specification documents were code named R1, R2, R3 as shown in the evaluation in Table II.

We use the following metrics to assess the performance of the system: Precision (P), Recall (R) and F-measure (F).

$$R = \frac{TP}{(TP+FN)} \quad P = \frac{TP}{TP+FP} \quad F = \frac{PR}{P+R}$$

In these formulas, TP, TN, FP and FN are as follows.

- *TP (true positive)*: number of requirements judged by both the expert and tool as being implicit
- *TN (true negative)*: number of requirements judged by both the expert and tool as not being implicit
- *FP (false positive)*: number of requirements judged by the tool as being implicit and by the expert as not being implicit
- *FN (false negative)*: number of requirements judged by the tool as not being implicit and by the expert as being implicit.

A group of subjects have been asked to assess the implicitness in the requirements documents and also use the tool. The subjects are a group of computing professionals, comprising software developers, academics and research students. They have been given the following instructions: 1) for each specified requirement, mark each requirement based on its implicit nature (noting that a requirement may contain more than one form of implicitness), and 2) For each requirement, specify the degree of criticality of each implicitness on a scale of 1 to 5 (1 being least critical to 5 being most critical). Table 1 shows a sample identification form used for this purpose. The type of implicitness includes: i) Ambiguity (A) ii) Incomplete Knowledge (IK) iii) Vagueness (V), and iv) Others (Miscellaneous).

Table 1: Sample Identification Form

S/N	Requirement	Type of Implicitness	Criticality				
1	The C&C shall provide the users with real-time data regarding the measured values, as collected from the various sensors part of the network.	(A)	1	2	3	4	5
		(IK)	1	2	3	4	5
		(V)	1	2	3	4	5
		(O)	1	2	3	4	5
2	The C&C shall support the configuration of ranges for sensor readings (maximum and minimum allowed values).	(A)	1	2	3	4	5
		(IK)	1	2	3	4	5
		(V)	1	2	3	4	5
		(O)	1	2	3	4	5
3	The C&C shall report potential sensor malfunctions to the users, when the reading is "Suspicious" or "Invalid".	(A)	1	2	3	4	5
		(IK)	1	2	3	4	5
		(V)	1	2	3	4	5
		(O)	1	2	3	4	5

5.3 Summary of Evaluation Results

Detailed evaluation has been conducted as described herewith. We summarize results of our evaluation, taking into account the feedback given by eight experts. Table 2 shows the recall, precision and F-scores computed for the tool relative to eight experts' (E1–E8) evaluations. For a detection tool, the recall value is definitely more important than precision. In the ideal case, the recall should be 100%, as it would relieve human analysts from the clerical part of document analysis [36]. For our tool with an average recall value of about 73.7%, the evaluation shows that the tool is fit for practical use, as it marks six out of eight IMR detected by humans and is consistent with best practices in software engineering. The average precision is 68.22% which shows the percentage of IMR judged by experts that was also retrieved by the tool is good and still consistent with software engineering best practices. The average F-score which is a harmonic mean of Precision and Recall is 70.3%. This shows that the result of the IMR support tool performance evaluation is very good and the tool is usable as determined by software engineers. As for

the IMRs marked by human evaluators but missed by the tool, manual examination has shown that they represent highly implicit factors where we could not identify explicit patterns that would allow the automation of IMR detection. This provides the potential for further research. Our observation from the simulation experiments is that the performance of the tool also depends significantly on the quality of the domain ontology and the CSKB, and the extent to which they are appropriately harnessed.

Table 2: Recall, Precision and F-Score metrics from 8 experts (E1-E8)

	Requirements	E1	E2	E3	E4	E5	E6	E7	E8	Average
Precision	R1	75	75	75	69.23	66.67	83.33	50	91.67	73.24
	R2	66.67	58.33	33.33	58.33	83.33	58.33	75	75	63.54
	R3	68.75	81.25	56.25	75	43.75	86.67	50	81.25	67.87
	Average									68.22
Recall	R1	90	90	75	90	80	83.33	75	78.57	82.74
	R2	66.67	70	66.67	58.33	76.92	70	69.23	75	69.1
	R3	73.33	72.22	64.29	66.67	58.33	81.25	61.54	76.47	69.26
	Average									73.7
F-Score	R1	81.82	81.82	75	78.26	72.73	83.33	60	84.62	77.2
	R2	66.67	63.63	44.44	58.33	80	63.63	72	75	65.46
	R3	70.97	76.47	60	70.59	50	83.87	55.17	78.79	68.23
	Average									70.3

6. Conclusions and Ongoing Work

In conclusion, the ability to automatically identify and manage IMRs will mitigate many risks that can adversely affect system architecture design and project cost in software development. This research involves a systematic IMR tool support framework which uses common sense knowledge that can be integrated into an organizational Requirements Engineering procedure for identifying and managing IMRs in software development processes. This is a direct response to problems in the practice of many organizations that have not been addressed by existing requirements management tools.

Thus, this work addresses the problem of identifying IMRs in Requirements documents and its further management. The novelty of this research is that integrates three core technologies, namely, common sense knowledge, ontology and text mining to propose an automated IMR framework. Another significant contribution is that an IMR support tool is developed based on the proposed framework and is helpful in domain-specific contexts. This has been evaluated and is found to yield good results that make the tool usable in practice as of now, as confirmed by the software engineering users.

Ongoing work includes conducting further research to enhance the performance of tool. We also aim to conduct detailed comparative studies with other state-of-the-art technologies in related areas. This potentially includes tools such as NAI, SR-Elicitor and ARUgen that can be used in IMR detection. We would consider different aspects such as lexical ambiguity, structural ambiguity and vagueness, using suitable performance evaluation metrics for comparison.

Our current and ongoing work would be useful to AI scientists and software engineers. Its targeted applications include providing software requirements specifications for various AI systems, where common sense is useful in automation. We particularly advocate the use of our proposed COTIR framework in the development of smart city tools in the future. This would fall under the general area of commonsense knowledge in smart cities. We envisage that it would have significant broader impacts.

Acknowledgments

This research was mainly conducted when Mr. Onyeka Emebo was a Visting PhD Scholar supported by the Fullbright Scholarship Program during academic year 2015-2016. He visited Montclair State University, NJ, USA from Covenant University, Ota, Nigeria where he is a PhD student with advisor Daramola Olawande. He worked at Montclair State University with advisor Aparna Varde from the Department of Computer Science. Early work in this research area started when Aparna Varde was a Visiting Researcher at the Max Planck Institute for Informatics in Saarbruecken, Germany in August 2015. We would like to mention the contributions of Niket Tandon from Allen Institute for Artificial Intelligence, Seattle, WA. His valuable inputs during this work and the permission for reuse of a relevant part of the WebChild code have been very helpful in this research activity. We sincerely thank all our sources of support.

References

- [1] A. Parameswaran,,: Capturing Implicit Requirements (02-08-2011), <http://alturl.com/emeej>
- [2] A. Stone, and P. Sawyer, : Identifying tacit knowledge-based requirements. In Software, IEE Proceedings-, vol. 153, no. 6, pp. 211-218. IET, 2006.
- [3] Zang, Liang-Jun, Cong Cao, Ya-Nan Cao, Yu-Ming Wu, and C. A. O. Cun-Gen. "A survey of commonsense knowledge acquisition." *Journal of Computer Science and Technology* 28, no. 4 (2013): 689-719.
- [4] Amgoud, L., Ouannani, Y., & Prade, H. Arguing by analogy—towards a formal view. A preliminary discussion. In *Working Papers of the 1st International Workshop on Similarity and Analogy-based Methods in AI (SAMAI'12)* (pp. 64-67).

- [5] C.D. Fellbaum, G.A. Miller (Eds.): WordNet: An Electronic Lexical Database. MIT Press, 1998.
- [6] Singh P, Lin T, Mueller E, Lim G, Perkins T, Zhu W. Open mind common sense: Knowledge acquisition from the general public. In Proc. Conf. Cooperative Information Systems, Oct.30-Nov.1 2002, pp.1223-1237.
- [7] Choi, F. Y. (2000, April). Advances in domain independent linear text segmentation. In Proceedings of the 1st North American chapter of the Association for Computational Linguistics conference (pp. 26-33). Association for Computational Linguistics.
- [8] Chowdhury, G. G. (2003). Natural language processing. Annual review of information science and technology, 37(1), 51-89.
- [9] Douglas B. Lenat: CYC: A Large-Scale Investment in Knowledge Infrastructure. Comm. of the ACM 38(11), pp. 32-38, 1995.
- [10] Decker, B., Ras, E., Rech, J., Klein, B., & Hoecht, C. (2005, November). Self-organized reuse of software engineering knowledge supported by semantic wikis. In Proceedings of the Workshop on Semantic Web Enabled Software Engineering (SWESE) (p. 76).
- [11] Fabbrini, F., Fusani, M., Gnesi, S., & Lami, G. (2001, June). An automatic quality evaluation for natural language requirements. In Proceedings of the Seventh International Workshop on Requirements Engineering: Foundation for Software Quality REFSQ (Vol. 1, pp. 4-5).
- [12] Gruninger, M., & Lee, J. (2002). Ontology-applications and design. Communications of the ACM, 45(2), 39-41.
- [13] Kamsties, E., Berry, D. M., Paech, B., Kamsties, E., Berry, D. M., & Paech, B. (2001, July). Detecting ambiguities in requirements documents using inspections. In Proceedings of the first workshop on inspection in software engineering (WISE'01) (pp. 68-80).

- [14] N. Tandon, G. de Melo, G. Weikum: Deriving a Web-Scale Common Sense Fact Database. AAAI 2011.
- [15] L. Singer, , O. Brill, , S. Meyer, , K. Schneider,: Utilizing Rule Deviations in IT Ecosystems for Implicit Requirements Elicitation. In: Proceedings of the Second International Workshop on Managing Requirements Knowledge (MaRK), pp. 22–26 (2009).
- [16] Lami, G., Gnesi, S., Fabbrini, F., Fusani, M., & Trentanni, G. (2004). An automatic tool for the analysis of natural language requirements. Informe técnico, CNR Information Science and Technology Institute, Pisa, Italia, Setiembre.
- [17] Lung, C. H., Urban, J. E., & Mackulak, G. T. (2007). Analogy-based domain analysis approach to software reuse. Requirements Engineering, 12(1), 1-22.
- [18] Maedche, A. (2012). Ontology learning for the semantic web (Vol. 665). Springer Science & Business Media.
- [19] Meyer, B. (1985). On formalism in specifications. IEEE software, 2(1), 6.
- [20] R. Speer, C. Havasi: Representing General Relational Knowledge in ConceptNet 5. LREC 2012.
- [21] O. Daramola, T. Moser, G. Sindre, and S. Biffl,: Managing Implicit Requirements Using Semantic Case-Based Reasoning Research Preview. REFSQ 2012, LNCS 7195, pp. 172–178, Springer-Verlag Berlin Heidelberg (2012).
- [22] R. Gacitua, B. Nuseibeh, , P. Piwek, , A.N. de Roeck, , M. Rouncefield, , P. Sawyer, , A. Willis, and H. Yang, “Making Tacit Requirements Explicit”, Second International Workshop on Managing Requirements Knowledge (MaRK'09) 2009.
- [23] R. Stoiber., M. Glinz, : Modeling and Managing Tacit Product Line Requirements Knowledge. Proceedings of the Second International Workshop on Managing Requirements Knowledge (MaRK'09), Atlanta, USA, September 2009.

- [24] G.E. Lebani, E. Pianta: Encoding Commonsense Lexical Knowledge into WordNet. Global WordNet Conference 2012.
- [25] Spanoudakis, G. (1996). Analogical reuse of requirements specifications: A computational model. *Applied Artificial Intelligence*, 10(4), 281-305.
- [26] Sugumaran, V., & Storey, V. C. (2002). Ontologies for conceptual modeling: their creation, use, and management. *Data & knowledge engineering*, 42(3), 251-271.
- [27] Welcome to Apache OpenNLP (24/10/2012) <http://opennlp.apache.org/>
- [28] Wilson, W. M., Rosenberg, L. H., & Hyatt, L. E. (1997, May). Automated analysis of requirement specifications. In *Proceedings of the 19th international conference on Software engineering* (pp. 161-171). ACM.
- [29] Yatskevich, M., & Giunchiglia, F. (2004). Element level semantic matching using WordNet. In *Meaning Coordination and Negotiation Workshop, ISWC*
- [30] Onyeka, E. (2013, May). A process framework for managing implicit requirements using analogy-based reasoning: Doctoral consortium paper. In *Research Challenges in Information Science (RCIS), 2013 IEEE Seventh International Conference on* (pp. 1-5). IEEE.
- [31] Tandon, N., de Melo, G., Suchanek, F., & Weikum, G. (2014). Webchild: Harvesting and organizing commonsense knowledge from the web. In *Proceedings of the 7th ACM international conference on Web search and data mining* (pp. 523-532). ACM.
- [32] Gharehchopogh, F. S., & Khalifelu, Z. A. (2011). Analysis and evaluation of unstructured data: text mining versus natural language processing. In *Application of Information and Communication Technologies (AICT), 2011 5th International Conference on* (pp. 1-4). IEEE.
- [33] Abma, B. J. M. "Evaluation of requirements management tools with support for traceability-based change impact analysis." Master's thesis, University of Twente, Enschede (2009).

[34] Software Requirements Specification – EMMON (12-07-2010)

<http://www.artemis-emmon.eu/deliverables/FP7-JU-EMMON-2010-DL-WP7-003-D7.1-software-requirements-specification-document.pdf>.

[35] TCS Software Requirements Specification (02-12-1999)

https://fas.org/irp/program/collect/docs/sss_20.pdf.

[36] Kiyavitskaya, N., Zeni, N., Mich, L., Berry, D.M.: Requirements for tools for ambiguity identification and measurement in natural language requirements specifications. *Requir. Eng.* 13 (2008) 207–239

[37] Varde, A., Tandon, N., Nag Chowdhury, S., Weikum, G.: Common Sense Knowledge in Domain Specific Knowledge Bases, Technical Report, Max Planck Institute for Informatics, Saarbruecken, Germany, August 2015.

[38] Du, X., Emebo, O., Varde, A., Tandon, N., Nag Chowdhury, S., Weikum, G.: Air Quality Assessment from Social Media and Structured Data: Pollutants and Health Impacts in Urban Planning, *IEEE International Conference on Data Engineering, ICDE, HDMM Workshop*, May 2016, Helsinki, Finland.